Lesgold, A.M., Resnick, L.B., & Beck, I.L. Preliminary results of a longitudinal study of reading acquisition. Paper presented at the annual meeting of the Psychonomic Society, San Antonio, November, 1978.

Lesgold, A.M., Roth, S.F., & Curtis, M.E. Foreground effects in discourse comprehension. Journal of Verbal Learning and Verbal Behavior, in press.

Riley, M.S. The development of children's ability to solve arithmetic word problems. Paper presented at the annual meeting of the American Educational Research Association, San Francisco, 1979.

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

Collaborations and medical use of programs via SUMEX

None.

Sharing and collaboration with other SUMEX-AIM projects

We have been in touch regularly with the ACT project members, and have agreed to test their new work with both simulations and by providing data in which both top-down-process-influencing variables and bottom-up-processing-influencing variables have been manipulated. Steven Roth has gathered such data. We have also been in touch with the group at Colorado [I don't know their project name], and are sharing information about possible future moves onto site-specific computer systems (see below) as well as the Kintsch discourse work, which relates to work we are doing (e.g., Lesgold, Roth, & Curtis, in press).

Critique of Resource Management

The system is getting crowded, but is certainly well managed. A critical issue for the future is the extent of decentralization and the proper role for a central resource. We see that role as one of communications among researchers and prototyping of work that does not yet merit a dedicated facility.

III. RESEARCH PLANS

Long range project goals and plans

Our plans are outlined in the above sections. We expect to have our own system at LRDC for INTERLISP use within the next year or so, depending upon progress within the lisp community in INTERLISP migration. Even then, it will be useful to have access to SUMEX to try out new ideas from other laboratories and to finish work that is hard to move (e.g., PLAS). However, in the long run, SUMEX will have been a major resource for a short period of time and a development tool for the next decade's work.

Justification and requirements for continued SUMEX use

As noted above, there is substantial work in progress that depends upon unique SUMEX resources, primarily the LISP systems. This, plus our interactions with Anderson, Kintsch, and and Polson, make SUMEX the desirable place for our work. In terms of publication and influence on the field, the record is public. We are in the process of securing a new system (with ARPA support), and will be moving off of SUMEX as fast as we can. But, it will be at least a year before the lisp resources we need are available.

Needs and plans for other computer resources beyond SUMEX-AIM

Discussed above.

Recommendations for future community and resource development

SUMEX must continue to move from being primarily a computer-power resource to being the leadership node for a group of users who are increasingly able to get their own computer power pretty cheaply. For example, the long-term future requires continued thought about networking schemes that will allow all of our 1981 lisp machines to be as collaborative as we are now in a central resource. This is where forthcoming work should center. Also, such tasks as INTERLISP and other system migration development might sensibly concentrate in SUMEX.

4.2 Stanford Projects

The following group of projects is formally approved for access to the Stanford aliquot of the SUMEX-AIM resource. Their access is based on review by the Stanford Advisory Group and approval by Professor Feigenbaum as Principal Investigator. As noted previously, the DENDRAL project was the historical core application of SUMEX. Although this is described as a "Stanford project," a significant part of the development effort and of the computer usage is dedicated to national collaborator-users of the DENDRAL programs.

4.2.1 AI Handbook Project

Handbook of Artificial Intelligence

E. A. Feigenbaum and A. Barr
Stanford Computer Science Department

I. SUMMARY OF RESEARCH PROGRAM

A. Technical Goals

The AI Handbook is a compendium of knowledge about the field of Artificial Intelligence. It is being compiled by students and investigators at several research facilities across the nation. The scope of the work is broad: Two hundred articles cover all of the important ideas, techniques, and systems developed during 20 years of research in AI. Each article, roughly four pages long, is a description written for non-AI specialists and students of AI. Additional articles serve as Overviews, which discuss the various approaches within a subfield, the issues, and the problems.

There is no comparable resource for AI researchers and other scientists who need access to descriptions of AI techniques like problem solving or parsing. The research literature in AI is not generally accessible to outsiders. And the elementary textbooks are not nearly broad enough in scope to be useful to a scientist working primarily in another discipline who wants to do something requiring knowledge of AI. Furthermore, we feel that some of the Overview articles are the best critical discussions available anywhere of activity in the field.

To indicate the scope of the Handbook, we have included an outline of the articles as an appendix to this report (see Appendix I on page 214).

B. Medical Relevance and Collaboration

The AI Handbook Project was undertaken as a core activity by SUMEX in the spirit of community building that is the fundamental concern of the facility. We feel that the organization and propagation of this kind of information to the AIM community, as well as to other fields where AI is being applied, is a valuable service that we are uniquely qualified to support.

C. Progress Summary

Because our objective is to develop a comprehensive and up-to-date survey of the field, our article-writing procedure is suitably involved. First drafts of Articles are reviewed by the staff and returned to the author (either an AI scientist or a student in the area). His final draft is then incorporated into a Chapter, which when completed is sent out for review to one or two experts in that particular area, to check for mistakes and omissions. After corrections and comments from our reviewers are incorporated by the staff, the manuscript is edited, and a final computer-prepared, photo-ready copy of the Chapter is generated.

We expect the Handbook to reach a size of approximately 1000 pages. Roughly half of this material will constitute Volume I of the Handbook, which will be going through the final stages of manuscript preparation in the Spring and Summer of 1979. The material in Volume I will cover AI research in Heuristic Search, Representation of Knowledge, AI Programming Languages, Natural Language Understanding, Speech Understanding, Applications-oriented AI Research, and Automatic Programming. Researchers at Stanford University, Rutgers University, SRI International, Xerox PARC, RAND Corporation, MIT, USC-ISI, Yale, and Carnegie-Mellon University have contributed material to the project.

D. List of Relevant Publications

The material in Volume I of the AI Handbook will appear in preliminary form as a Stanford Computer Science Technical Report in the Summer of 1979, and will be published shortly thereafter.

II. INTERACTIONS WITH SUMEX-AIM RESOURCE

A. Collaborations and medical use of programs via SUMEX

We have had a modest level of collaboration with a group of students and staff at the Rutgers resource, as well as occasional collaboration with individuals at other ARPA net sites.

B. Sharing and interactions with other SUMEX-AIM projects.

As described above, we have had moderate levels of interaction with other members of the SUMEX-AIM community, in the form of writing and reviewing Handbook material. During the development of this material, limited arrangements have been made for sharing the emerging text. As final manuscripts are produced, they will be made available to the SUMEX-AIM community both as on-line files and in the hardcopy, published edition.

C. Critique of Resource Management

Our requests of the SUMEX management and systems staff, requests for additional file space, directories, systems support, or program changes, have been answered promptly, courteously and competently, on every occasion.

III. RESEARCH PLANS (8/78 - 7/81)

A. Long Range Project Goals

The following is our tentative schedule for completion and publication of the AI Handbook:

Spring and Summer, 1979 - Volume I will go through final editing, computer typesetting, and printing.

Fall, 1979 through Spring, 1980 - Volume I will be published. Research for Volume II will be started and draft material will go through the external review process.

Summer, 1980 - Final editing, typesetting and publication of Volume II.

B. Justifications and requirements for continued SUMEX use

The AI Handbook Project is a good example of community collaboration using the SUMEX-AIM communication facilities to prepare, review, and disseminate this reference work on AI techniques. The Handbook articles currently exist as computer files at the SUMEX facility. All of our authors and reviewers have access to these files via the network facilities and use the document-editing and formatting programs available at SUMEX. This relatively small investment of resources will result in what we feel will be a seminal publication in the field of AI, of particular value to researchers, like those in the AIM community, who want quick access to AI ideas and techniques for application in other areas.

C. Your needs and plans for other computational resources

We use document preparation facilities (the XEROX Graphics Printer) at the Stanford AI Laboratory. We are also planning to use the new TEX typesetting system being developed by Prof. D. Knuth of the Stanford Computer Science Department.

D. Recommendations for future community and resource development

None.

4.2.2 AGE - Attempt to Generalize

AGE - Attempt to Generalize

H. Penny Nii and Edward A. Feigenbaum
Computer Science Department
Stanford University

ABSTRACT: Isolate inference, control, and representation techniques from previous knowledge-based programs; reprogram them for domain independence; write a rule-based interface that will help a user understand what the package offers and how to use the modules; and make the package available to other members of the AIM community and labs doing knowledge-based systems development, and the general scientific community.

I. SUMMARY OF RESEARCH PLAN

Technical Goals

The general goal of the AGE project is to demystify and make explicit the art of knowledge engineering. It is an attempt to formulate the knowledge that knowledge engineers use in constructing knowledge-based programs and put it at the disposal of others in the form of a software laboratory.

The design and implementation of the AGE program is based primarily on the experience gained in building knowledge-based programs at the Stanford Heuristic Programming Project in the last decade. The programs that have been, or are being, built are: DENDRAL, meta-DENDRAL, MYCIN, HASP, AM, MOLGEN, CRYSALIS [Feigenbaum 1977], and SACON [Bennett 1978]. Initially, the AGE program will embody artificial intelligence methods used in these programs. However, the long-range aspiration is to integrate methods and techniques developed at other AI laboratories. The final product is to be a collection of building-block programs combined with an "intelligent front-end" that will assist the user in constructing knowledge-based programs. It is hoped that AGE will speed up the process of building knowledge-based programs and facilitate the dissemination of AI techniques by: (1) packaging common AI software tools so that they need not be reprogrammed for every problem; and (2) helping people who are not knowledge engineering specialists write knowledge-based programs.

Medical Relevance and Collaboration

AGE is relevant to the SUMEX-AIM Community in two ways: as a vehicle for disseminating cumulated knowledge about the methodologies of knowledge engineering and as a tool for reducing the amount of time needed to develop knowledge-based programs.

1. Dissemination of Knowledge: The primary strategy for conducting AI research at the Stanford Heuristic Programming Project is to build complex programs to solve carefully chosen problems and to allow the problems to condition the choice of scientific paths to be explored. The historical context in which this methodology arose and summaries of the programs that have been

built over the last decade at HPP are discussed in [Feigenbaum 1977]. While the programs serve as case studies in building a field of "knowledge engineering," they also contribute to a cumulation of theory in representation and control paradigms and of methods in the construction of knowledge-based programs.

The cumulation and concomitant dissemination of theory occur through scientific papers. Over the past decade we have also cumulated and disseminated methodological knowledge. In Computer Science, one effective method of disseminating knowledge is in the form of software packages. Statistical packages, though not related to AI, are one such example of software packages containing cumulated knowledge. AGE is an attempt to make yesterday's "experimental technique" into tomorrow's "tool" in the field of knowledge engineering.

2. Speeding up the Process of Building Knowledge-based Programs: Many of the programs built at HPP are intelligent agents to assist human problem solving in tasks of significance to medicine and biology (see separate sections for discussions of work and relevance). Without exception the programs were handcrafted. This process often takes many years, both for the AI scientists and for the experts in the field of collaboration.

AGE will reduce this time by providing a set of preprogrammed inference mechanisms that can be used for variety of tasks. Close collaboration is still necessary to provide the knowledge base, but the system design and programming time of the AI scientists can be significantly reduced. Since knowledge engineering is an empirical science, in which many programming experiments are conducted before programs suitable for a task are produced, reducing the programming and experimenting time would significantly reduce the time required to build knowledge-based systems.

Progress Summary

SUMMARY OF CURRENT USAGE

Currently AGE-1 is available to a limited number of groups on the PDP-10 at the SUMEX-AIM Computing Facility. In the process of building AGE, we have used it to write some programs: CRYPTO, a program that solves cryptogram problems [Aiello 1979]; two different versions of PUFF [Feigenbaum 1977; Kunz 1978]--one using the Event-driven control macro and another using the Expectation-driven control macro [Nii 1978]. Since the domain-specific knowledge for PUFF already existed and was being used in EMYCIN, the AGE version took about a week to bring up--time needed to reorganize the existing rules into KSs and to rewrite the rules in AGE rule syntax. Psychologists from the University of Colorado (see a separate description within the Pilot Projects Section) have begun to experiment with AGE in order to develop research programs in reading comprehension and design processes. It was also used by a person outside the AGE project to write a knowledge-based program for a part of the game of hearts [Quinlan 1979].

Profile Of The Current AGE System

To correspond to the two general technical goals described earlier, AGE is being developed along two separate fronts: the development of tools and the development of "intelligent" user interface.

Currently Implemented Tools

The current AGE system provides the user with a framework useful for incremental hypothesis formation, known as the Blackboard Model [Erman 1975; Lesser 1977]. The framework, with which the user builds his Blackboard-based program, has been implemented to allow flexibility in representation and in the application of other problem solving methods within the framework. It consists of three major components:

1. The Blackboard: The blackboard contains hypotheses in a hierarchical data structure; it represents the task domain in terms of a hierarchy of analysis levels of the task.

2. The KSs: The KSs contain the knowledge of the task domain (which the user must provide) that can perform the analysis. The KSs are represented as sets of production rules [Davis 1977].

3. The Control: The control component contains mechanisms that allow the user to (a) specify the conditions for the invocation of the KSs and (b) to select items on the blackboard for focus of attention.

A paper by Nii [1979] describes in somewhat more detail the current implementation of the Blackboard framework in AGE.

The "Intelligent" Front-end in AGE

Currently the "intelligence" in the front-end is limited to: (a) a tutorial subsystem that allows the user to browse through the textual knowledge base, and (b) a design subsystem that guides the user through each step of program specification.

Tutor Subsystem: The textual knowledge base contains (a) a general description of the building-block components at the conceptual level, (b) a description of the implementation of these concepts within AGE, (c) a description of how these components are to be used within the user's program, (d) how they can be constructed by the user, and (e) various examples. The information is organized in a network to represent the conceptual hierarchy of the components and to represent the functional relationship among them.

Design Subsystem: The knowledge necessary for AGE to guide the user in design and construction is represented in a data structure in the form of an AND/OR tree that. It represents, on one hand, all the possible structures available in the current AGE system; and, on the other hand, represents the decisions the user must make in order to design his program. Using this schema, the design subsystem guides the user from one design decision point to another. At each decision point, the user has access to the textual knowledge base, to advice on the decisions to be made at that point, and to acquisition functions that aid the user in specifying the appropriate component.

A paper by Aiello [1979] contains an extensive example of the various interactions currently possible in AGE.

II. RESEARCH PLAN

Research Topics

The task of building a software laboratory for knowledge engineers is divided into two main sub-tasks:

1. The isolation of techniques used in knowledge-based systems. It has always been difficult to determine if a particular problem solving method used in a knowledge-based program is "special" to a particular domain or whether it generalizes easily to other domains. In existing knowledge-based programs, the domain specific knowledge and the manipulation of such knowledge using AI techniques are often so closely coupled that it is difficult to make use of the programs for other domains. One of our goals is to isolate the AI techniques that are general and determine precisely the conditions for their use.

2. Guiding the user in the initial application of these techniques. Once the various techniques are isolated and programmed for use, an intelligent agent is needed to guide the user in the application of these techniques. Initially, we assume that the user understands AI techniques, knows what she wants to do, but does not understand how to use the AGE program to accomplish her task. A longer range interest involves helping the user determine what techniques are applicable to her task, i.e. we will assume that the user does not understand the necessary techniques of writing knowledge-based programs. Some immediate questions to be posed are: Is there a "best way" to represent knowledge that would apply to many task domains? Is there a flexible data representation that could describe many types of objects? What is the best way to handle differences in the ability of the users of the AGE program?

Research Plan

Version 1 of the AGE program is now complete. Next step in the research and development plan includes the following:

1. Improving the Front-end

Tutor Subsystem: Although the current textual knowledge base is organized to provide explanation in various forms, the program is not intelligent enough to know what the user does not understand. The problems involved in providing an intelligent tutorials are similar to those in Intelligent CAI. AGE will track the research in this area and improve the Tutor subsystem.

Design Subsystem: Although the current Design subsystem provides acquisition functions that allow the user to interactively specify the knowledge of the domain and control structure, it does not (aside from simple advice) provide the user any help in the designing process. For example, AGE should be able to provide some heuristics on what kinds of inference mechanisms and representation are appropriate for different kinds of problems. We have begun collecting knowledge-engineering heuristics, but much more work is needed in building a design-aid system that will be useful to the user.

2. Adding More Tools

Our concept of a software laboratory is a facility by which the users are provided with a variety of preprogrammed problem-solving frameworks--similar in spirit to designs of prefabricated houses. The user augments and modifies a framework to develop his own programs. At the same time, we need to provide the user with diverse tools. We currently have a framework for developing user programs that use the Blackboard framework. We are currently adding a framework for backward-chained inference mechanisms. Another inference mechanism, the heuristic search paradigm, will be added. We will also integrate parts of Unit Package and add the capability for representing knowledge as a semantic network.

References

Aiello, N. and Nii, H.P., "Building a knowledge-based system with AGE," Stanford Heuristic Programming Project Memo HPP-79-3, 1979.

Bennett, J., Creary, L., Engelmore, R., Melash, R., "SACON: A knowledge-based consultant for structural analysis," Heuristic Programming Project Memo HPP-78-23, 1978.

Davis, R. and King J., "An overview of production systems," Machine Intelligence 8: Machine Representation of Knowledge, Elcock, E.W. and Michie, D. (eds.), John Wiley, 1977.

Engelmore, R.S. and Nii, H.P., "A knowledge-based system for the interpretation of protein x-ray crystallographic data," Stanford Heuristic Programming Project Memo HPP-77-2, January, 1977.

Erman, L.D. and Lesser, V.R., "A multi-level organization for problem solving using many, diverse, cooperating sources of knowledge," Proc. 4th IJCAI, 1975, pp.483-490.

Feigenbaum, Edward A., "The art of artificial intelligence: I. Themes and case studies of knowledge engineering," Proc. IJCAI 5, 1977, pp.1014-1029.

Kunz, J.C., Fallat, R.J., McClung, D.H., Osborn, J.J., Votteri, B.A., Nii, H.P., Aikins, J.S., Fagan, L.M., Feigenbaum, E.A., "A physiological rule based system for interpreting pulmonary function test results," Heuristic Programming Project Memo HPP-78-20, (submitted to Computers and Biomedical Research), 1978.

Lesser, V.R. and Erman, L.D., "A retrospective view of the HEARSAY-II architecture," Proc. 5th IJCAI, 1977, pp. 790-800.

Nii, H.P. and Aiello, N., AGE (Attempt to Generalize): Profile of the AGE-0 System, Stanford Heuristic Programming Project Memo HPP-78-5 (Working paper), June 1978.

Nii, N. Penny and Aiello Nelleke, AGE (Attempt to Generalize): "A knowledge-based program for building knowledge-based programs," Stanford Heuristic Programming Project Memo HPP-79-4, also to appear in the Proc. of IJCAI 6, 1979.

Quinlan, J. R., "A knowledge-based system for locating missing high cards in bridge," to appear in the Proc. of IJCAI 6, 1979.

4.2.3 DENDRAL Project

Resource-Related Research - Computers In Chemistry
The DENDRAL Project

Prof. Carl Djerassi
Department of Chemistry
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

I.A. Technical Goals

In the second year of our current grant we have continued to address the problems of computer-assisted structure elucidation and the applications of the resulting programs to biomedical structure elucidation. We have focussed our attention on development of interactive computer programs which are designed to act as chemists' assistants in exploration of the potential structures for unknown compounds. These programs take into account structural information derived from a variety of sources including both physical and chemical methods. We are extending the interpretative power of these programs to enable them to draw meaningful structural conclusions from chemical data. To meet these objectives we are developing a series of computer programs, described in more detail below, which emulate several important aspects of manual approaches to structure elucidation.

I.B. Medical Relevance and Collaboration

Chemical structure elucidation is a problem to common to many efforts throughout the biomedical community. Knowledge of chemical structure is a necessary first step to further study of properties relevant to biomedicine, such as those involved in pharmacology or toxicology. Our instrumentation and computer programs have been directed specifically both to development of new techniques to assist in biomolecular structure elucidation and to applications to a number of structural problems in our own group and the research groups of our collaborators. As our research has matured, we have been able to move toward computer representation and manipulation of three-dimensional representations (specifically, configurational stereoisomerism at this time) of molecular structure. Recent and proposed developments will provide computer assistance in a wide range of problems which require this level of structural description, for example, relationships of structure to observed properties such as biological activities. We have paid particular attention during the past year to the dissemination and availability of our programs to large segments of the biomedical community. These collaborative efforts are described in detail in Section II.

II.C. Progress Summary

Reprogramming CONGEN

We previously discussed a preliminary effort to reprogram the structure generator algorithm of CONGEN into the BCPL language. This early experience showed BCPL to be a compact and efficient language containing all of the basic features needed for the full reprogramming effort. Continued development has produced a version of CONGEN in BCPL which contains nearly all of the features of the INTERLISP/SAIL/FORTRAN version. The primary exception is the perception of aromaticity, and this feature is currently being implemented. The BCPL version has the following advantages over the previous one;

a) It requires less than 10% as much computer memory, due partly to the more compact coding and partly to the use of an overlay structure;

b) It uses about 2-5 times less computer time on typical cases than the most highly-tuned (block compiled) previous version of CONGEN;

c) The redesigned front-end provides significantly more error checking, a simpler and more flexible input format, and a more thorough "help" facility;

d) It can easily deal with problems an order of magnitude larger.

e) It is exportable to a variety of different computers.

Overlay Structure

As portions of CONGEN were developed in BCPL, estimates of its eventual size could be made and it became obvious that the entire program would occupy a somewhat larger amount of memory (about 150 K words as compared to the roughly 450 K words for the earlier version) than is usually available to individual users at many installations (on the order of 50-60 K words at TOPS-10 sites). Because the processing in CONGEN falls naturally into several independent activities (generating intermediate structures, imbedding, defining substructures, etc.), the program can easily be broken into separate overlay segments which need to communicate only relatively small amounts of information. In the interest of transportability, though, it was decided not to rely upon the overlay mechanism provided by any particular operating system or language. The safest approach seemed to be to divide the overall program into completely independent, separately runnable modules capable of starting one another and communicating with one another via disk files. The drawback of this approach is that there may be a significant overhead in creating and reading files, and in switching from one module to another. But because all information needed to describe a CONGEN session is maintained on file, the program is unusually robust; even if an error causes the program to crash, CONGEN can simply be restarted and it will restore the complete environment which existed before the erroneous command was issued. Also, a particular operating system may offer some means of accomplishing overlays efficiently, and by interfacing the modules through a small control program, it should be possible to take advantage of such facilities. Under TENEX on the PDP-10, for example, a program may control a large number of forks (independent virtual address spaces) each containing a

separate program. We have successfully interfaced the CONGEN modules through a small fork-manipulating program so that the overhead of starting a particular module is paid only once for each CONGEN session.

The current CONGEN is composed of eight modules, the largest of which occupies about 46 K words of memory and the rest of which fall in the range 15-36 K words. We are exploring ways of reducing the size of this largest module (SURVEY - see below) to bring it into this range also. The modules and their functions are as follows:

a) CONGEN (35 K) - Main control module, user interaction, error checking;

b) EDITS (19 K) - User interaction for defining substructures;

c) GENERA (26 K) - Generation of intermediate structures;

d) PRUNE (15 K) - Elimination of structures based on structural features;

e) IMBED (36 K) - Expansion of superatoms in intermediate structures;

f) DRAW (26 K) - Output of structural drawings to the user's terminal;

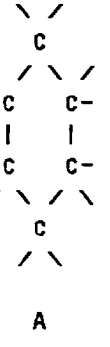
g) SURVEY (46 K) - Examination of large structure lists for frequency of occurrence of standard structural features; and

h) STEREO (24 K) - Generation of stereoisomers.

CONGEN Developments

The reprogramming effort has been far from a transliteration of existing algorithms into BCPL. In many portions, the basic algorithmic approach taken in the previous version was reformulated to allow for a more effective representation and solution of the problem. The major milestones in CONGEN development which have paralleled the reprogramming are as follows:

1) Imbedder. The mathematical technique for expanding superatoms in intermediate structures developed by Brown was reexamined and reformulated to allow for a more compact representation. The primary difference in our new approach is that the topological symmetry group of the atoms, rather than the free valences, is used in the computation. For example, the superatom A below, with twelve free valences, has twelve topological symmetry operations



A

interchanging its atoms, but because of the pairwise interchanges between free valences on each atom, the free-valence group has 64x12=768 symmetry elements. The BCPL version of the imbedder carries the symmetry information as 11 permutations of 6 objects (the identity permutation is not explicitly represented) requiring 66 words of memory, rather than as 768 permutations of 12 objects requiring 9216 words of memory. By implicitly representing interchange symmetry among free valences, among the termini of internal bonds being allocated to the superatom and among monovalent atoms being attached to the superatom, the new version is able to use a drastically smaller amount of space for the storage of symmetry information.

Neither of these approaches to imbedding can perceive all possible sources of duplicate structures, so it was necessary also to develop a final filter package to canonicalize the imbedded structures and compare them for duplicates. However, the new version stores the structure representations on an external random-access file rather than in the computer's memory as was done before, and only a list of pointers to these filed structures is stored internally. As a result, the new imbedder can deal with thousands rather than hundreds of imbedded structures using only a modest amount of memory.

2) Constraints. The basic structure generation and imbedding algorithms are of little practical use without the ability to constrain their output based on the presence or absence of structural features. The graph matcher and cycle finder, which accomplish constraint testing, were translated with little change from their INTERLISP counterparts. Inclusion of constraints in the imbedder, where they serve only as a filter on the final output structures, was straightforward. In the structure generator, however, the constraint-testing mechanism was merged much more intimately with the generation process. The main aspects of this merging are as follows:

a) As soon as hydrogen atoms are distributed among the non-hydrogen atoms (the first activity of the generator), the distributions are checked against the constraint substructures to determine which distributions can be ruled out a priori. If a substructure is required to be present and contains three methine carbons (CH), for example, the generator will immediately discard hydrogen distributions which do not contain at least three such carbons. Many constraints supplied to the generator place restrictions on the possible distributions of hydrogen atoms, and by this mechanism such constraints are tested most efficiently.

b) The order in which the generator assembles its atoms is influenced by which atoms appear in the constraints. If a substructure forbidding the construction of peroxides (O-O) is present, the generator will be encouraged to consider possible interconnections among oxygen atoms first so that the presence of peroxides can be avoided early in the computation. Because different constraints may encourage different starting atoms, a scoring scheme has been developed which is used to establish the overall order of atom assembly, taking all constraints into account.

3) Interactive aids. Much effort has been directed toward the development of a robust and helpful interactive system to allow a user easily to define a CONGEN problem and to make use of the basic algorithmic tools. The primary accomplishments in this direction have been as follows:

a) The development of LINSTR, a package of BCPL functions for interactive input from the user, accessed by all of the interactive CONGEN modules. The line-input and prompting functions in LINSTR provide for three levels of help information which can easily be passed from the main program. The first level consists of prompts which are typed to the user when information is required by the program. The novice may step through the prompting sequences supplying one piece of information at a time in response to these prompts, while the expert user may anticipate the prompts and type ahead his responses on the line to avoid the prompts. This, together with the ability of the LINSTR functions to accept unambiguous abbreviations for keywords, allows a great deal of flexibility in the form of the input. For example, the following two sequences accomplish the same effect in the program (user's responses underlined):

Step-by-step input;

```
 DEFINE
DEFINITION TYPE: SUBSTRUCTURE
NAME: R6
(NEW SUBSTRUCTURE)
>RING 6
>DONE
R6 DEFINED
```

Condensed input;

```
 DE S R6;R 6;DO
(NEW SUBSTRUCTURE)
R6 DEFINED
```

A second level of help is provided by the '?' facility which can can be evoked at any prompt in the program. At these points, the '?' input will cause helpful information passed by the main program to LINSTR to be typed to the user. The third level of help is provided by a similar '??' facility, which will cause the program to refer to a much more extensive on-line help document to give a full description of the expected information, and the context in which it will be used. This third level is still under development; the basic mechanism has been developed but we have not yet constructed the on-line documentation.

b) The simplification and extension of the basic commands. The number of basic CONGEN commands has been reduced from 29 to 14 by the consolidation of commands with similar function (e.g., SHOW is now a general purpose method of obtaining information about the session and replaces six previous commands) and eliminating little-used options (e.g., TREEGEN). The number of EDITSTRUC commands has likewise been reduced from 23 to 17. Also, previous concepts which were somewhat artificial have been removed. For example, a user does not now need to distinguish between superatoms and patterns when he defines a substructure. The representations for these two types of substructure have been consolidated and a defined substructure can be used in either

context. As another example, the user does not need to place substructures on BADLIST any more - the new input sequence allows him to express the presence or absence of substructural features in a natural statement such as 'exactly 3' or 'at most 1' or 'none'. The new command structure seems easier for users to remember and work with. Our experience in the workshops held at Stanford (see below), which were attended by many persons totally unfamiliar with computers and interactive systems, indicated that the new interface between scientist and the machine was much simpler to use and represented a major improvement over the old version of CONGEN to which some of the persons had been exposed previously.

Stereochemistry

The first version of the stereoisomer generator program was written in SAIL and has been improved in several ways. The program has been modified to process lists of structures to count and/or generate the possible stereoisomers. Thus with the existing CONGEN structure generator it is now possible for the first time to generate all the possible stereoisomers for a given empirical formula completely and irredundantly. These stereoisomers are represented in a compact canonical form and are written onto a disk file by the program along with other information about the structure. Three additional features which were proposed in the last annual report have been added to this program. First, at the user's discretion, the program will compute cis and trans double bond designations for the stereoisomers and write these on the file. Second R and S designations for tetravalent stereocenters based on the Cahn-Ingold-Prelog conventions are computed for stereocenters which are not fixed by any nontrivial symmetry element. These designations were thought to be the most useful and most stable with respect to future changes of the R/S nomenclature system. Third, the ability to handle stereochemistry of common heteroatoms with valence less than 5 has been added. A small interactive package has been added for deciding whether trivalent nitrogen atoms are free to invert. The user is given a choice for each such nitrogen atom.

This program has been included with the current LISP version of CONGEN (it runs as a separate fork) and is available to all users who can access SUMEX. It has been extensively tested on well over 1000 structures.

Since the CONGEN program has been recently reprogrammed into BCPL to create an exportable version, it was decided to also reprogram the STEREO program into BCPL and carry on further developments in that language to ensure compatibility and exportability. With the exception of the parts of the program which compute R/S symbols and handle heteroatoms interactively, this reprogramming has been accomplished. Further developments on this program include a fairly extensive interactive package which allows the user to obtain information about the generated stereoisomers. The user may obtain drawings of projected stereocenters showing absolute configurations of stereocenters (e.g., Fischer projections, Newman projections, double bonds) or obtain drawings of linear segments of structures showing all the configurations of the included stereocenters. The user may also obtain information about the symmetry and equivalent atoms in any stereoisomer. This program is currently running with the BCPL version of CONGEN and was available and tested during the recent series of workshops. This program has been exported with this version of CONGEN.

The experimental version of the BCPL program has been modified to allow for some constrained generation of stereoisomers The algorithm and program for exhaustive generation were written with this eventuality in mind. An additional interactive session has been added to the stereoisomer generator which allows the user to add constraints before generating the stereoisomers. At present, the user may input constraints on the absolute or relative stereochemistry of any stereocenters. Thus if part of the stereochemistry of a structure is known, it is possible to constrain the stereoisomer generator to produce just those isomers consistent with the known stereochemistry. This parallels the procedure in the structure generator of CONGEN.

Structure checking functions for CONGEN

A program, "STRUCC", has been developed to provide functions for checking sets of structures for desired substructural features or for compatibility with recorded mass-spectral or nmr data. While primarily devised for processing sets of structural isomers produced by means of CONGEN , STRUCC can also take as input sets of structures created through the REACT program or defined though an extension of CONGEN's EDITSTRUC function.

The main structure checking functions currently available through STRUCC are:

1) EXAMINE: This EXAMINE function is an extended version of that available in standard CONGEN. Amongst other extensions are facilities for checking for specified ring-fusions or spiro-junctions within structures.

2) MSA: The MSA ("Mass Spectral Analysis") functions provide a means for using mass spectral data to rank candidate structures. The MSA functions can employ either ordinary "half-order theory", or a model of fragmentation in which bond break plausibilities are related to specified substructural features.

3) LOOK: The LOOK functions are intended to assist a user in investigating the utility of proposed experiments for differentiating between candidate structures. LOOK provides a mechanism for determining the various different ways in which particular superatom parts are incorporated into candidate structures.

4) TSYM: The TSYM function allows some simple forms of symmetry constraint to be defined. These constraints use only topological symmetry.

5) RESONANCECHECK: The RESONANCECHECK function is intended for checking that all constraints have been given to the structure generator. The function can identify differences in candidate structures that would be associated with features in the 1Hnmr or 13Cnmr that one might reasonably expect to be fairly obvious (e.g. different numbers of hydroxy protons, different numbers of carbonyl carbons etc). Generally, such differences are found in cases where the user has forgotten to specify substructural features incompatible with the observed data, or has misapplied the constraints so that not all instances of unwanted features are eliminated.

6) NMRFLT: The NMRFLT functions represent a first attempt at developing a system for predicting proton resonance spectra of candidate structures, and for using differences between predicted and observed spectra as a basis for pruning the structure list.

The STRUCC system is also used as a test-bed for new structure evaluation functions. When functions are considered to be sufficiently developed to be of use, top-level calls to those functions are added to STRUCC's repertoire of commands.

STRUCC has a user-interface similar to that of CONGEN and incorporates many of the same subsystems (e.g. EDITSTRUC and DRAW).

Meta-DENDRAL

INTSUM - The INTSUM program for the analysis of spectra has been improved by using confidence factors in the place of many of the original program constraints. This feature allows association of likelihoods with fragmentations. It thus allows consideration of a much wider range of possible processes while limiting the final explanations for spectrum peaks to the most plausible explanations.

Additional improvement of the program allows logical separation of the concepts of H-transfers and neutral composition transfers. This provides a better correlation between the explanations provided by the program and those expected by the chemist.

RULEGEN - A significant problem in generalizing the INTSUM explanations has always been reducing the size of the search space so as to be able to produce interesting rules in a reasonable amount of time. In addition to the constraints already provided, the RULEGEN program now allows use of existing rules to filter the peak explanations to be considered. This is an important step in allowing the program to focus on rules which account for peak explanations not yet encompassed by existing rules. As an aid in better understanding the process of rule formation, the program is now capable of generating additional information about the search space. This information serves as data for other programs which can then analyze and present to the user compact descriptions of the rule search done by RULEGEN.

EDITSTRUC INTERFACE - The latest versions of the structure editor, EDITSTRUC, and the structure drawing programs have been interfaced to allow their use in all appropriate places in INTSUM and RULEGEN. The newest programs for conversion of EDITSTRUC structures recognize a larger subset of the structural features which may be specified within EDITSTRUC. This allows the user greater flexibility in the specification of substructures in user-created rules.

PREDICTION and RANKING - The programs allowing the entry and use of user-defined rules have been extended to allow prediction of the molecular ion and inclusion of confidence factors in the rules.

The process of spectrum prediction from Meta-DENDRAL rules has previously involved the matching of rules against only those sites in the molecules considered as possible breaks. With the use of user-entered rules, and program

developed rules containing greater structural detail, the program was generalized to allow prediction based on graph matching alone, without the prior generation of possible break sites.

HUMAN ENGINEERING - Many minor improvements have been made in the program's interaction with the user. In general, these improvements have been designed according to the following criteria: 1. Messages should be informative yet not excessively long or wordy; 2. User typing should be kept to a minimum; 3. Programs should behave in ways which people find understandable; 4. During execution, programs should provide occasional information concerning their progress.

RESULTS - The practical value and capability of new programs are best evaluated by applying them to real, non-trivial problems. In our case, we have chosen the biologically important marine sterol compounds. Their mass spectra are predominant in the structure elucidation of new compounds in spite of the fact that relatively few of the fragmentation mechanisms are known. Often very similar spectra are recorded due to the great similarity of common skeletons.

Our study involves the comparison of predicted spectra of known structures with the observed spectra of unknown compounds. We want to compare the usefulness of different methods of forming the rules used for spectrum prediction. We distinguish 3 methods:

1) Half-order theory (can be supplemented by functional group rules);

2) Class-specific rules (selected by the chemist);

3) Computer-generated rules.

Our results were obtained using nine selected 4-demethylsterols (six isomers of composition C29H48O, two C28H46O and one C27H44O). Each spectrum of the nine selected marine sterols was considered to be the observed spectrum and ranked against 23 candidate structures (the 23 candidates contained 17 different C7 - C11 sidechains and three 4-demethylsterol skeletons). For the half-order theory an overall average performance of (2.4 0.9) was obtained. The first number gives the number of candidates ranked better than the correct one, the second represents the number of candidates ranked equally with the correct one. In this case the average value is not very representative, as its value is strongly reduced by a compound which was ranked in 17th place. This compound, the 23-demethylgorgosterol, contains a cyclopropane in the sidechain for which no special fragmentation processes are considered in the simple half-order theory. The ranking can be greatly improved by providing fragmentation rules for cyclopropane rings.

The results of the second method (class specific rules), depends on the quality and number of selected rules. For this study we selected about 17 skeleton breaks (observed in more then 70 percent of the structures) from the INTSUM results of 23 marine sterols to which we added 13 known fragmentation processes. These processes (associated with neutral transfers, intensity range, and a confidence factor) were entered using the new rule editor program. The overall performance of these rules was (0.3 0) which means that, with the exception of three compounds, which were ranked in the second position, the

correct structure was always ranked first. A further improvement is seen when the distribution of the scoring values is considered. For these rules, much better separations were observed than with the half-order theory. Also, the quality of the predicted spectra are sufficient to consider the creation of a library which could be visually compared without the need of a computer. For the third method no results can be summarized here, as the computer generated rules are still being developed. The improvement of this last step will be a main goal of the next year.

MAXSUB Program - The function of the MAXSUB program is to detect common structural features in a potentially diverse but related set of compounds. This problem is one faced by chemists engaged in structure/activity studies, particularly in design of new, biologically active compounds based on known compounds with known activities. However, any problem involving an "activity" related to structure, including spectral signatures, is in principle amenable to analysis by MAXSUB. MAXSUB, by determining common features of structures displaying common activities, is presumably focussing on those aspect of the structures which are related to the activity. However, in its current state, the program is only experimental. Many types of activity are intimately connected with stereochemical aspects of structure and MAXSUB does not include any stereochemistry. It does represent a foundation for further study of the problem because the algorithms can in principle deal with three-dimensional descriptors of atoms and bonds. Some work may be done on this program in the next grant period.

GENOA Program - The GENOA program (for GENeration with Overlapping Atoms) has been developed from a preliminary INTERLISP version to overcome one of the most serious deficiencies of the current CONGEN program. This deficiency is that substructures and atoms input to CONGEN's structure generator must not overlap. This condition is probably the most serious stumbling block to the chemist's interpretation of partial structural information for the CONGEN program. The GENOA program overcomes this problem by allowing structures to overlap in all possible ways. This program removes the need for the chemist to interpret his data in terms of nonoverlapping substructures and enhances the "intelligence" of the program as a chemist's assistant.

Briefly, GENOA obtains the molecular formula for an unknown compound, and the number (may be an integer, a range or zero) and name of inferred substructures, one at a time. For each new substructure, GENOA builds the requested number and ensures that the required number of all previous substructures is met. Utility functions allow definition of substructures, and visualizing and saving all intermediate results. Substructural statements are simply made to GENOA. The program determines not only how the required substructures can be built, but also makes structural inferences concerning the implications of each statement.

After the last known substructure is specified, a simplified structure generator, not the one utilized in CONGEN, is used currently to build complete structures (alternatively the problem could be saved at this stage and additional substructural information supplied at a later time, continuing the problem where it was left off). The generator is quite inefficient and creates many duplicate structures which must be removed (automatically). Control is then passed directly to the CONGEN program where all the currently available utilities for

further processing, e.g., STEREO, MSANALYZE, may be used to prune or explore further the structural candidates.

II.D. List of Relevant Publications

(1) T.H. Varkony, R.E. Carhart, and D.H. Smith, "Computer-Assisted Structure Elucidation. Modelling Chemical Reaction Sequences Used in Molecular Structure Problems," in "Computer-Assisted Organic Synthesis," W.T. Wipke, Ed., American Chemical Society, Washington, D.C., 1977, p. 188.

(2) "Computer-Assisted Structure Elucidation," D.H. Smith, Ed., American Chemical Society, Washington, D.C., 1977.

(3) R.E. Carhart, T.H. Varkony, and D.H. Smith, "Computer Assistance for the Structural Chemist," in "Computer-Assisted Structure Elucidation," D.H. Smith, Ed., American Chemical Society, Washington, D.C., 1977, p. 126.

(4) D.H. Smith, M. Achenbach, W.J. Yeager, P.J. Anderson, W.L. Fitch, and T.C. Rindfleisch, "Quantitative Comparison of Combined Gas Chromatographic/Mass Spectrometric Profiles of Complex Mixtures," Anal. Chem., 49, 1623 (1977).

(5) B.G. Buchanan and D.H. Smith, "Computer Assisted Chemical Reasoning," in "Computers in Chemical Education and Research," E.V. Ludena, N.H. Sabelli, and A.C. Wahl, Eds., Plenum Press, New York, N.Y., 1977, p. 401.

(6) D.H. Smith and R.E. Carhart, "Structure Elucidation Based on Computer Analysis of High and Low Resolution Mass Spectral Data," in "High Performance Mass Spectrometry: Chemical Applications," M.L. Gross, Ed., American Chemical Society, 1978, p. 325.

(7) T.H. Varkony, D.H. Smith, and C. Djerassi, "Computer-Assisted Structure Manipulation: Studies in the Biosynthesis of Natural Products," Tetrahedron, 34, 841 (1978).

(8) D.H. Smith and P.C. Jurs, "Prediction of 13C NMR Chemical Shifts," J. Am. Chem. Soc., 100, 3316 (1978).

(9) T.H. Varkony, R.E. Carhart, D.H. Smith, and C. Djerassi, "Computer-Assisted Simulation of Chemical Reaction Sequences. Applications to Problems of Structure Elucidation," J. Chem. Inf. Comp. Sci., 18, 168 (1978).

(10) D.H. Smith, T.C. Rindfleisch, and W.J. Yeager, "Exchange of Comments: Analysis of Complex Volatile Mixtures by a Combined Gas Chromatography-Mass Spectrometry System," Anal. Chem., 50, 1585 (1978).

(11) W.L. Fitch, P.J. Anderson, and D.H. Smith, "Isolation, Identification and Quantitation of Urinary Organic Acids," J. Chrom., 162, 249 (1979).

(12) W.L. Fitch, E.T. Everhart, and D.H. Smith, "Characterization of Carbon Black Adsorbates and Artifacts Formed During Extraction," Anal. Chem., 50, 2122 (1978).

(13) W. L. Fitch and D. H. Smith, "Analysis of Adsorption Properties of Adsorbed Species on Commercial Polymeric Carbons," Environ. Sci. Tech., 13, 341 (1979).

(14) J.G. Nourse, R.E. Carhart, D.H. Smith, and C. Djerassi, "Exhaustive Generation of Stereoisomers for Structure Elucidation," J. Am. Chem. Soc., 101, 1216 (1979).

(15) C. Djerassi, D.H. Smith, and T.H. Varkony, "A Novel Role of Computers in the Natural Products Field," Naturwiss., 66, 9 (1979).

(16) N.A.B. Gray, D.H. Smith, T.H. Varkony, R.E. Carhart, and B.G. Buchanan, "Use of a Computer to Identify Unknown Compounds. The Automation of Scientific Inference," Chapter 7 in "Biomedical Applications of Mass Spectrometry," G.R. Waller, Ed., in press.

(17) T.C. Rindfleisch and D.H. Smith, in Chapter 3 of "Biomedical Applications of Mass Spectrometry," G.R. Waller, Ed., in press.

(18) T.H. Varkony, Y. Shiloach, and D.H. Smith, "Computer-Assisted Examination of Chemical Compounds for Structural Similarities," J. Chem. Inf. Comp. Sci., in press.

(19) R. Carlson, S. Popov, I. Massey, C. Delseth, E. Ayanoglu, T.H. Varkony, and C. Djerassi, Bioorg. Chem., 7, 453 (1978).

(20) J.G. Nourse, "The Configuration Symmetry Group and its Application to Stereoisomer Generation, Specification and Enumeration," J. Am. Chem. Soc., 101, 1210, (1979).